Adaptive Integration and Approximation over hyper-rectangular regions with applications to basket options pricing

Christophe de Luigi, Sylvain Maire

Adaptive integration and approximation over hyper-rectangular regions with applications to basket options pricing

Christophe De Luigi, Sylvain Maire
Université du Sud Toulon-Var, I.S.I.T.V.
Avenue G. Pompidou BP 56, F-83162 La Valette du Var CEDEX, FRANCE
{deluigi,maire}@univ-tln.fr

December 18, 2009

Abstract

We describe an adaptive algorithm to compute piecewise sparse polynomial approximations and the integral of a multivariate function over hyper-rectangular regions in medium dimensions. The key ingredient is a quasi-Monte Carlo quadrature rule which can handle the numerical integration of both very regular and less regular functions. Numerical tests are performed on functions taken from Genz package in dimensions up to 5 and on basket options pricing.

1 Introduction

This paper deals with an adaptive numerical computation of the integral and the approximation of a multivariate function over an hyper-rectangular region in dimension s.

Numerical integration over hyper-rectangles and especially over the hypercube $D = [0,1]^s$ has been extensively studied. The non-adaptive numerical integration methods are of two main types. The first one relies on quadrature formulae based on points uniformly distributed in D. This concerns Monte Carlo (MC) [8] and Quasi-Monte Carlo (QMC) integration [16] where the positive quadrature weights are equal and also quantization [18] where they are not. Their accuracy depend on respectively variance, discrepancy and distortion which are not too much sensitive to the regularity of the integrand and to the dimension s.

The second type is constituted of quadrature formulae based on interpolation or approximation on different reduced size bases. Korobov spaces [9] have been introduced in the case of $s-$dimensional Fourier bases on periodic functions.

They are built using that the Fourier coefficients a_m verify

$$|a_m| \leq \frac{C}{(\widetilde{m_1}\widetilde{m_2}...\widetilde{m_Q})^\gamma}$$

where $\widetilde{m} = \max(1, |m|)$ and $\gamma > 1$ is linked to the regularity of the integrand. The corresponding quadrature formulae are lattice rules [9,22] which try to annihilate the most significant Fourier coefficients according to this decay. Non-periodic functions can be periodised [6] to still use these quadratures but with an increasing constant of decay. In the case of polynomial approximations, Novak and Ritter [17] have obtained exact quadrature formulas on spaces of polynomials such that the total degree is below a given value. Based on the use of the control variates method on piecewise interpolation polynomials, Atanassov and Dimov [1] built a numerical method reaching an optimal rate of convergence for multivariate smooth functions belonging to a space where the total degree of differentiation is fixed.

In many situations, the function to integrate or approximate may have completely different behaviours in terms of variations or even in terms of regularity in different parts of the domain D. In those cases, it might be more efficient to split adaptively D in subregions according to error indicators based on the quadrature points. The most famous adaptive MC integration routines are Miser [19] and Vegas [11]. They rely respectively on stratified and importance sampling and error indicators based on the empirical variance. Quasi-Monte Carlo versions of these two algorithms have been introduced in [20]. A very complete comparison between these new algorithms and usual ones based on hierachical quadratures [5] has been done in [21] on functions taken from Genz package [4].

Our algorithm relies on the quadrature formulae developed in [15] which conciliates the two types of integration methods as they are built by fitting QMC drawings to a reduced Tchebychef polynomial model inspired of Korobov spaces. In section 2, we remind the constuction and the properties of these formulae. We discuss various adaptive strategies [21] based on hierarchical quadrature formulae for the integral but also for some of the other coefficients of the multivariate polynomial approximation. In section 3, we test these strategies on functions taken from Genz package from dimension 2 to 5. We try to find out which strategy is the most efficient for the different functions of this package. Finally in section 4, we study the pricing of basket options [2] in dimensions 2 to 4 for which we give accurate estimations up to 10 digits of calls and puts validated using the parity call-put formula as no exact values exist.

2 Description of the adaptive algorithm

2.1 Quasi-Monte Carlo quadratures

Whenever one wants to develop an adaptive method, he should first choose an efficient quadrature rule in the non-adaptive case. While MC or QMC methods

can be used for integrands with no or few regularity, usual quadrature rules are designed for regular functions in general and sometimes only for functions having a given type of regularity. It is worth having quadratures that are simultaneously efficient for very regular and less regular functions. These quadratures should also not be too sensible to the dimensional effect. The quadratures developed in [15] have such nice properties as they combine the approximation on reduced Tchebychef polynomial basis and the use of QMC points to compute this approximation. They are especially efficient for very smooth functions but can also handle pretty well continuous functions. They have been obtained after successive improvements of an initial adaptive MC method [12] via quasi-random sequences [14] and the introduction of Tchebychef polynomial basis of Korobov type [13]. We summarize the construction of these formulae for a multivariate function in the hypercube $[-1,1]^s$. Letting $\widehat{m} = \max(1,m)$, we introduce the set

$$W_{s,d} = \{m \in \mathbb{N}^s / \, (\widehat{m_1}...\widehat{m_s}) \leq d\}$$

which corresponds to a level d of approximation. We can then give the reduced Tchebychef polynomial approximation

$$f(x_1,x_2,..,x_s) \simeq \sum_{m \in W_{s,d}} b_m T_{m_1}(x_1) T_{m_2}(x_2)..T_{m_s}(x_s)$$

where the $L_{s,d} = card(W_{s,d})$ coefficients b_n are the weighted mean-square coefficients. The definition of $W_{s,d}$ is motivated by the fact that we have proved in [13] that

$$|b_m| \leq \frac{C_1}{(\widehat{m_1}\widehat{m_2}...\widehat{m_Q})^{2L}}$$

for a C^{2L} function. This enables us to select a priori the leading coefficients in a standard tensor product mean-square approximation of f. Once this selection is done, it remains to compute a numerical approximation of the coefficients b_n. This is obtained by fitting the model

$$\sum_{m \in W_{s,d}} b_m T_{m_1}(x_1) T_{m_2}(x_2)..T_{m_s}(x_s)$$

to the observations of the function f at points $P_i = (X_i^{(1)},...,X_i^{(s)})$ with $1 \leq i \leq N$. The choice of the points P_i is crucial for the condition number $\kappa(A)$ of the least-square matrix A. We have proved in [15] that taking these points as independent random variables with density the multidimensional Tchebychef weight

$$w(x_1,x_2,..,x_s) = \prod_{i=1}^{s} \frac{1}{\pi\sqrt{1-x_i^2}} 1_{[-1,1]}(x_i)$$

is a very good choice. Indeed, in this case $\kappa(A)$ goes to 1 when $N \to \infty$ at a MC speed $\frac{\sigma}{\sqrt{N}}$, where the variance σ^2 is always bounded by one whatever the set $W_{s,d}$ is. An even better choice is to use QMC points or points obtained

via quantization [18] to represent the density w. Once the matrix A is built, its inverse is computed numerically, which enables to build quadrature formulae for each of the coefficients b_m and finally a quadrature formula for the integral itself. In the following, the quadrature points are built using $\beta \times L_{s,d}$ Halton points with additional points very close to each corner of the domain for a total of $M = \beta \times L_{s,d} + 2^s$ points. In most situations, the parameter β is chosen equal to 3 which is sufficient to ensure a small $\kappa(A)$. The corner points are control points to detect a possible change of regularity in the function f. The quadrature formula

$$Q_{s,d,M}(f) = \sum_{i=1}^{M} \alpha_i f(X_1^{(i)}, ..., X_s^{(i)})$$

is the approximation of

$$I(f) = \int_{[-1,1]^s} f(x)dx$$

and $Q_{s,d,M,b_m}(f)$ is the corresponding approximation of the coefficient b_m. The same quadrature formulae on a given rectangle R are $Q_{s,d,M,R}(f)$ and $Q_{s,d,M,b_m,R}(f)$.

2.2 Adaptive strategies

2.2.1 Error indicators

Two main things are crucial for an adaptive method to be efficient: the error indicator and the splitting strategy. Error indicators can be of different types, based on the empirical variance or on hierachical quadratures. We have at our disposal many error indicators as we are able to build a whole family of quadrature formulae for the integral of f but also for some of the coefficients of its weighted mean-square approximation. Moreover, the residual in the least-square approximation is also an error indicator. We have made many numerical experiments for finally choosing hierarchical estimators based on two quadrature formulae. We select two sets W_{s,d_1} and W_{s,d_2} with $1 \leq d < d_2$, the corresponding quadrature formulae $Q_{s,d_1,M}(f)$ and $Q_{s,d_2,M}(f)$ for the integral of f and also the quadrature formulae for some of the leading coefficients in the approximation model. These coefficients are the $s+1$ coefficients belonging to the set A_s of the coefficients b_m for which all the indexes $(m_1, m_2.., m_s)$ are equal to zero or only one is non-zero and equal to one. Our error indicator $E_{s,d_1,d_2,R}(f)$ is

$$|Q_{s,d_1,M,R}(f) - Q_{s,d_2,M,R}(f)| + \sum_{m/b_m \in A_s} |Q_{s,d_1,M,b_m,R}(f) - Q_{s,d_2,M,b_m,R}(f)|$$

on a given rectangle R. This indicator is more robust than the usual indicator based on the comparison between only $Q_{s,d_1,M,R}(f)$ and $Q_{s,d_2,M,R}(f)$. Indeed, it may happen in some situations that these last values are very close to each other but are both wrong. This is less likely to happen for all the estimators

of the leading coefficients. This indicator is also a lot cheaper than the one based on the residual of the least-square approximation as we compute only few coefficients of this approximation. It made no difference in terms of accuracy in using this indicator or the residual. The approximate value of the integral on a given rectangle is $Q_{s,d_2,M,R}(f)$.

2.2.2 Splitting strategy

We now describe the splitting strategies used in the numerical experiments. In all methods, at each step of the algorithm, the list of all the hyper-rectangles involved in the algorithm is stored and these hyper-rectangles are sorted according to their error indicator. The first strategy is the fully adaptive (FA) strategy. It consists in trying the s possible ways to divide the hyper-rectangle R with largest error indicator in two equal size pieces R_1 and R_2 along one of the axis and keeping only the best splitting. The best splitting is the one for which

$$E_{s,d_1,d_2,R_1}(f) + E_{s,d_1,d_2,R_2}(f)$$

is minimum among the s possible choices. Another strategy called non-adaptive (NA), even if it is, consists in dividing the hyper-rectangle also in two equal size pieces R_1 and R_2 uniformly at random among the s possible directions. Finally, we can make a mix between the two strategies by doing first some steps of the FA method to find the regions of higher interest and then the other steps by the NA method for the sake of robustness. Whatever the method is, when the splitting is done, rectangle R is removed from the list and the rectangles R_1 and R_2 are inserted in the list according to their error indicators $E_{s,d_1,d_2,R_1}(f)$ and $E_{s,d_1,d_2,R_2}(f)$. We can also mention the method developed in [3] and applied to financial mathematics where the number of rectangles and also their locations are optimized adaptively.

3 Numerical experiments on the Genz package

3.1 Description of the Genz package

To make comparisons between different numerical integration methods, one needs to have various examples of functions with significantly different behaviours and also among these different sort of functions to define clearly the degree of difficulty of the numerical integration procedure. Each family of functions should be large enough to make statistics on the correct digits of integration methods possible and obviously the exact values of each integral should be known. Genz package [4] gives examples of such functions which are of 6 different types and each of them depends of affective vector parameters a and unaffective vector parameters u. These functions as well as their degree of difficulty measured by $\|a\|_1$ are listed below. The integral to compute is $I(f) = \int_{[0,1]^s} f(x)dx$.

Oscillatory $f_1(x) = \cos(2\pi u_1 + \sum_{i=1}^{s} a_i x_i)$, $\|a\|_1 = \frac{110}{\sqrt{s^3}}$

Product peak $f_2(x) = \prod_{i=1}^{s} \frac{1}{a_i^{-2}+(x_i-u_i)^2}$, $\|a\|_1 = \frac{600}{s^2}$
Corner peak $f_3(x) = (1+\sum_{i=1}^{s} a_i x_i)^{-(s+1)}$, $\|a\|_1 = \frac{600}{s^2}$
Gaussian $f_4(x) = \exp(-\sum_{i=1}^{s} a_i^2 (x_i-u_i)^2)$, $\|a\|_1 = \frac{100}{s}$
Continuous $f_5(x) = \exp(-\sum_{i=1}^{s} a_i |x_i-u_i|)$, $\|a\|_1 = \frac{150}{s^2}$
Discontinuous $f_6(x) = \exp(\sum_{i=1}^{s} a_i x_i) 1_{x_1<u_1, x_2<u_2}$, $\|a\|_1 = \frac{100}{s^2}$

The degree of difficulty increases with $\|a\|_1$. For example, when $\|a\|_1$ increases the function f_1 oscillates more and the variance in the Gaussian function f_4 is getting smaller which makes its peak harder to detect. The parameters u_i are uniform independent random variables in $[0,1]$ and the vector $a = (a_1, a_2, .., a_s)$ is obtained by scaling uniform independent random variables in $[0,1]$ such that the difficulty based on $\|a\|_1$ is met. To evaluate the efficiency of an integration routine for a given type of functions, a number M of random functions are drawn and the average number of correct digits is computed. The adaptive method may fail to converge. In this case, it might be preferable to give the worst case error or other statistical estimators like the median to measure its performances.

3.2 Fully adaptive method in dimension two and three

Even though numerical integration routines are often tested in high dimensions, it is worth testing already their accuracy on the hard examples of the Genz package in dimension $s = 2$ or $s = 3$. We test the FA method with two different error indicators, the first one based on $E_{s,12,15,R}(f)$ and the second one based on $E_{s,5,8,R}(f)$. In dimension two, we plot in figures (1)-(4) for the functions f_1, f_2, f_4 and f_5 the meshes obtained using the first indicator after 100 steps of the algorithm on one successful random example for each of these functions.

For the function f_1, the method is essentially non adaptive as we observe a mesh close to a uniform one. For the functions f_2 and f_4, the mesh concentrates where these functions have the larger variations: around the origin for f_2 and around the Gaussian peak for f_4. The mesh tends to detect the two lines of discontinuity of the function f_5. We give now quantitative examples based on 20 random samples of each function of the Genz package. We denote by $I_{k,N}(f)$ the approximation of the integral $I_k(f)$ of the kth random function after N steps of the algorithm. We compute numerically in table 1 the number of correct digits

$$D_{k,N} = -\frac{\log|I_{k,N}(f) - I_k(f)|}{\log|I_k(f)|}$$

and denote by $\bar{D}_N$ and $\tilde{D}_N$ its average and median over the 20 random functions. For the method based on $E_{s,12,15,R}(f)$, we compute these estimators after $N_1 = 100$ and $N_2 = 1000$ steps. The corresponding number of function evaluations used in the algorithm is $N_{tot} = N \times (3 \times L_{2,15} + 2^2) \times 2 = N \times 232$. To make a fair comparison between the two methods, the number of steps is adapted to the ratio $\frac{L_{2,15}}{L_{2,8}}$ so that the number of function evaluations is the same. This leads to number of steps $M_1 = 202$ and $M_2 = 2018$ for the algorithm based on

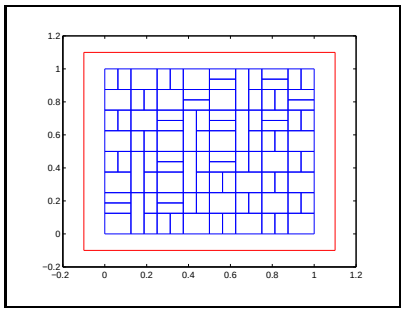

Figure 1: Mesh for f_1

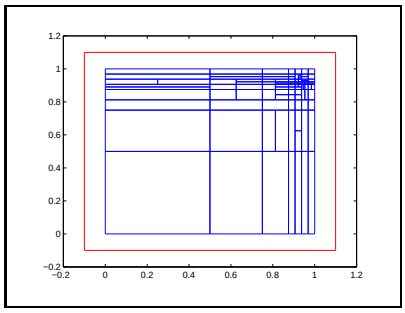

Figure 2: Mesh for f_2

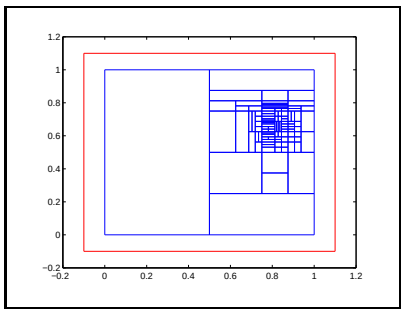

Figure 3: Mesh for f_4

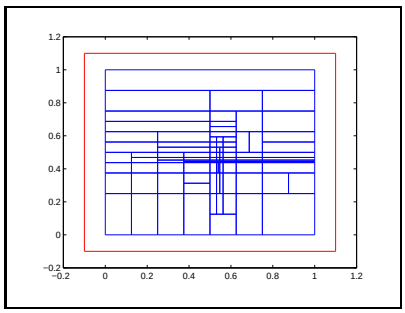

Figure 4: Mesh for f_5

$E_{s,5,8,R}(f)$. We also compare with the mean value over the 20 random functions using QMC integration based on $N_2 \times (3 \times L_{2,15} + 2^2) \times 2 = 232000$ Halton points.

For all functions but f_6, the FA method is very accurate and outperforms QMC integration. For the functions f_1 and f_3, median and mean are very close, which means that the algorithm always converges. For the functions f_4, f_5 and especially f_2, mean and median are significantly different which shows convergence problems. For example, 5 times out of 20 the accuracy was less than 3 digits on the integral of f_2. Except for function f_5, it was always better to use a higher degree formula with a smaller number of iteration steps. Finally, the method is not efficient at all for the discontinuous function f_6. We now look at the same examples in dimension 3 (see table 2) to see if the same conclusions hold and how the method is sensitive to the dimensional effect. The maximum number of evaluation points is now 836000.

For all functions, the method is less accurate than in dimension 2. It remains a lot better than QMC integration for functions f_1, f_3 and f_4. The accuracy is still poor for the discontinuous function f_6. For function f_2, high degree formulae are required to have a good accuracy. Finally, the method is only

	$\check{D}_{M_1}$	$\bar{D}_{M_1}$	$\check{D}_{M_2}$	$\bar{D}_{M_2}$	$\check{D}_{N_1}$	$\bar{D}_{N_1}$	$\check{D}_{N_2}$	$\bar{D}_{N_2}$	QMC
f_1	6.1	6.2	10.9	10.8	7.5	7.4	12.1	12.2	2.9
f_2	4.8	4.1	9.9	7.1	6.5	6.1	10.7	9.5	3.3
f_3	7.9	8.1	11.5	11.6	8.4	8.5	12.5	12.6	1.8
f_4	6.9	6.5	11.7	11	9.2	8.8	13.4	12.7	3.7
f_5	5.7	5.5	9.8	9	5.4	5.5	7.1	6.8	4.4
f_6	2.3	2.5	2.6	2.7	2.6	2.6	2.8	2.8	3.2

Table 1: Accuracy in dimension 2

	$\check{D}_{M_1}$	$\bar{D}_{M_1}$	$\check{D}_{M_2}$	$\bar{D}_{M_2}$	$\check{D}_{N_1}$	$\bar{D}_{N_1}$	$\check{D}_{N_2}$	$\bar{D}_{N_2}$	QMC
f_1	5	4.8	7.4	7	6.8	7	10.5	10.8	3.2
f_2	2.2	2.2	2.6	3	5.1	4.5	8.4	7.1	4.1
f_3	4.9	5.2	7.9	8.1	6.4	6.6	9.5	9.3	1.9
f_4	4.7	4.3	8.2	6.8	6.4	6.3	10	9.7	4.2
f_5	4.2	3.9	4.3	4.2	5	4.9	5.7	5.9	5.3
f_6	2.1	2.2	2.4	2.6	2.2	2.1	2.2	2.1	3.9

Table 2: Accuracy in dimension 3

slightly better than QMC for f_5. As explained in [21], this function is getting harder to integrate because the number of hyperplanes of non-differentiability increases with s.

3.3 Non-adaptive methods

Using an adaptive method based on quadrature formulae, Schurer [21] has obtained on function f_6 a better accuracy than with QMC. Our FA method is poor for this function. In order to improve it, we change the quadratures and the splitting strategy. First, we take more Halton points to build our quadratures for a fixed level d of accuracy. Instead of taking $3 \times L_{s,d} + 2^s$ integration points, we now take $20 \times L_{s,d} + 2^s$ points in order to be more accurate for non-regular functions. To be more robust, the splitting strategy can be changed in the NA one described in section 2.2.2. In figure 5, we compare five different methods on f_6 in dimension 2: the FA method based on $E_{2,5,8,R}(f_6)$ and $E_{2,12,15,R}(f_6)$, the NA based on the same error indicators and QMC integration. We can see that the FA method has convergence problems as the accuracy stops to increase after some steps of the algorithm. Its version based on more quadrature points is nevertheless more accurate. The NA method is more accurate than QMC especially with the error indicator $E_{2,12,15,R}(f_6)$. We can conclude that the NA method can handle the problems posed by discontinuous functions. It has also been tested on the other functions giving good results but is in general less accurate than the FA method. It can be recommended when one is looking for

robustness in the numerical integration procedure.

3.4 Higher dimensions

We now give further comparisons in dimensions 4 and 5 using the FA method for the regular functions (f_1 to f_4) of the package. As noticed in section 3.2, higher degree formulae seem to be more efficient for the regular functions and especially for f_2. For this function, we compare, in figure 6, 4 quadrature formulae built using respectively $E_{4,5,8,R}(f_2)$, $E_{4,20,25,R}(f_2)$, $E_{4,30,35,R}(f_2)$ and Halton points. We observe that the higher the degree of the formulae is the more accurate the algorithm is. Furthermore, we can notice that our algorithm based on $E_{4,30,35,R}(f_2)$ becomes better than the QMC method when the number of integrand evaluation increases while this method is known to be very efficient on this particular function. We now keep the error indicator $E_{s,30,35,R}(f_i)$ for $s = 4, 5$ and plot the accuracy for the 4 functions using our algorithm. In dimension 4 (see figure 7), the order of all the methods (given by the slope of the curves) is near two. This leads to a maximum accuracy between 7 digits for f_2 and 13 digits for f_1. In dimension 5 (see figure 8), the order of the method for f_1 and f_3 is still near two but is now near one for f_2 and f_4. The maximum accuracy is still 13 digits for function f_1 but near 6 or 7 for the other functions. Note that we obtain a two digits better accuracy than with the QMC version of MISER developed in [20] on the tough examples of functions f_2 and f_3 in dimension 5.

4 Pricing basket options

4.1 Basket options

An european call (put) option [10] consists in paying now a certain price to have the right to buy (sell) at a fix date T an asset S_T at a given value K called the strike price. If we assume that the model is Black-Scholes with an interest rate r and a volatility σ than the price $V(T,K)$ of a call option is

$$\mathrm{E}[(S_T-K)_+] = \exp(-rT)\int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi}}\exp(-\frac{y^2}{2})(S_0\exp((r-\frac{\sigma^2}{2})T+\sigma y)-K)_+dy$$

This value can be computed exactly as a closed form exists so no numerical integration problem for the moment. Denoting by $U(T,K) = \mathrm{E}[(K-S_T)_+]$ the value of the put option, we have the parity call-put option formula [10]

$$V(T,K) - U(T,K) = S_0 - K\exp(-rT).$$

A basket options is a generalization of an european option to multiple assets. For two assets, we have for example

$$S_t = a_1 S_t^{(1)} + a_2 S_t^{(2)}$$

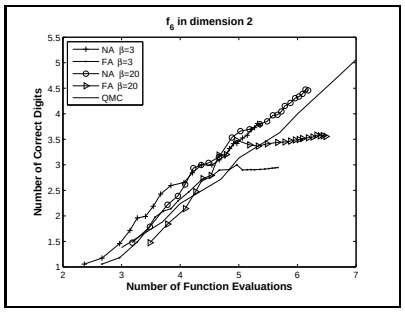

Figure 5: f_6 in dim 2

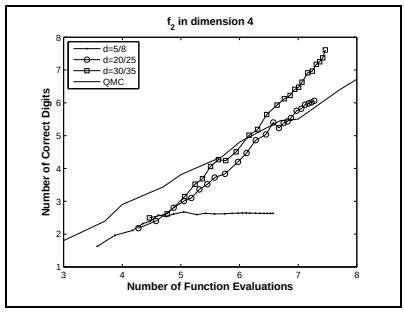

Figure 6: f_2 in dim 4

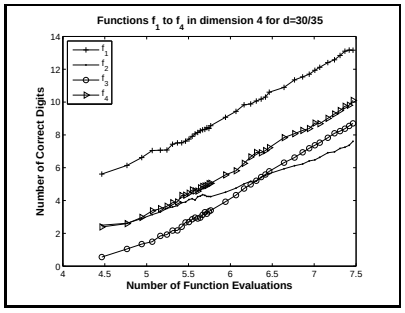

Figure 7: f_1, f_2, f_3 and f_4 in dim 4

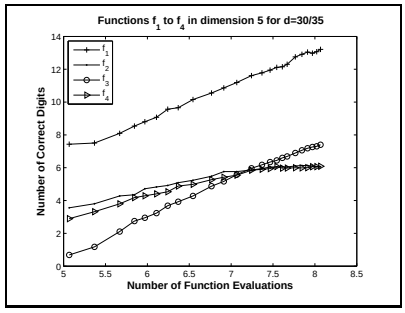

Figure 8: f_1, f_2, f_3 and f_4 in dim 5

where a_1 and a_2 are the proportions of the assets $S_t^{(1)}$ and $S_t^{(2)}$ in S_t. The assets are in general correlated and the price of the call option $V(T,K) = \mathrm{E}[(S_T - K)_+]$ in a Black-Scholes model is

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{2\pi}\exp(-\frac{(x^2+y^2)}{2})(K\exp(-rT) - \psi(x,y))_+ dxdy$$

where

$$\psi(x,y) = a_1\exp(-\frac{\sigma_1^2}{2}T + \sigma_1 x) + a_2\exp(-\frac{\sigma_2^2}{2}T + \sigma_2 r_0 x + (1-r_0^2)^{0.5}\sigma_2 y)$$

and where σ_1, σ_2, r_0 are the parameters of the correlation matrix. The parity call-put formula now writes

$$V(T,K) - U(T,K) = a_1 + a_2 - K\exp(-rT).$$

In general, this formula is used to obtain the hardest to compute (in terms of variance) between the call or the put option price once the other has been computed numerically. Here, we compute the call and the put option prices independently and use this formula as a criterion of accuracy. The infinite

integral is truncated and we let

$$V(T,K,B) = \int_{-B}^{B}\int_{-B}^{B} \frac{1}{2\pi}\exp(-\frac{(x^2+y^2)}{2})(K\exp(-rT) - \psi(x,y))_+ dxdy$$

the truncated estimation of $V(T,K)$ and $U(T,K,B)$ the analogous estimation of $U(T,K)$. For q assets, we have

$$S_t = a_1 S_t^{(1)} + a_2 S_t^{(2)} + .. + a_q S_t^{(q)}$$

and hence q-dimensional integrals to compute for the option prices.

4.2 Numerical examples

4.2.1 Examples in dimension 2

We consider two numerical examples similar as in [2] with the parameter values

$$T = 3, r = 0.05, a_1 = a_2 = 50, r_0 = 0.3, \sigma = \sigma_1 = \sigma_2 = 0.4$$

for the first example and

$$T = 3, r = 0.05, a_1 = a_2 = 50, r_0 = 0.7, \sigma = \sigma_1 = \sigma_2 = 0.2$$

for the second one. Three different values for the strike price are tested, one at the money $K_1 = 100$, one out $K_2 = 127.80$ and one completely out $K_3 = 300$. Two thousands steps of the FA method are used with quadrature formulae of degrees 15 and 20 which corresponds to a number of function evaluations of $2000 \times 325 \times 2 \simeq 1.2 \times 10^6$. We give in table 3, the values of the truncated estimations for $B_1 = 12$ and $B_2 = 13$. We also compute

$$C(T,K,B) = |V(T,K,B) - U(T,K,B) - a_1 - a_2 + K\exp(-rT)|$$

for these reference values and denote by $H(T,K,B)$ the same error indicator but based on computations of the price of the basket options using a QMC method with 1.2×10^6 Halton points.

$\sigma = 0.4, r_0 = 0.3$	$V(T,K,B)$	$U(T,K,B)$	$C(T,K,B)$	$H(T,K,B)$
$(K,B) = (K_1,B_1)$	28.49407707	14.56487473	1×10^{-8}	7×10^{-4}
$(K,B) = (K_1,B_2)$	28.49407711	14.56487474	1×10^{-8}	6×10^{-4}
$(K,B) = (K_2,B_1)$	18.85549193	28.85397134	2×10^{-8}	8×10^{-4}
$(K,B) = (K_2,B_2)$	18.85549196	28.85397133	7×10^{-9}	6×10^{-4}
$(K,B) = (K_3,B_1)$	1.810536589	160.0229295	2×10^{-8}	9×10^{-4}
$(K,B) = (K_3,B_2)$	1.810536598	160.0229295	4×10^{-10}	1×10^{-3}

Table 3: Option pricing: example 1

On this first example, we obtain a very good accuracy of at least 8 digits on all the computations of the prices of call and put options. Indeed the values we

$\sigma = 0.2, r_0 = 0.7$	$V(T,K,B)$	$U(T,K,B)$	$C(T,K,B)$	$H(T,K,B)$
$(K,B) = (K_1, B_1)$	20.04091112	6.111708770	3×10^{-10}	4×10^{-4}
$(K,B) = (K_1, B_2)$	20.04091112	6.111668676	2×10^{-6}	3×10^{-4}
$(K,B) = (K_2, B_1)$	8.915343222	18.91382261	3×10^{-9}	5×10^{-4}
$(K,B) = (K_2, B_2)$	8.915343226	18.91382261	2×10^{-10}	4×10^{-4}
$(K,B) = (K_3, B_1)$	0.021755879	158.2341488	2×10^{-10}	9×10^{-4}
$(K,B) = (K_3, B_2)$	0.021755880	158.2341488	6×10^{-11}	7×10^{-4}

Table 4: Option pricing: example 2

obtain are both validated by the parity call-put formula and by the comparison between the truncated approximations for the two different values of B.

The second example confirms the efficiency of the algorithm on an example with a smaller variance and more correlated assets (see table 4). The accuracy is even better than on the first example except when $(K,B) = (K_1, B_2)$ where the value of $U(T, K_1, B_2)$ is less accurate (but with still 6 correct digits) certainly due to some convergence problems. On both examples, the accuracy of the QMC method is at most 4 digits.

It is also interesting to plot the meshes obtained for call or put options. We plot in figures 9 and 10 these meshes for $(K,B) = (K_1, B_2)$ in the first example and in figures 11 and 12 for $(K,B) = (K_3, B_2)$ in the second example. We observe that the refinement is done mainly near the frontier separating the region where the function vanishes and the region where it is positive. This frontier is obviously the same for the call and the put option as observed numerically. If it crosses the origin for the at the money options, it is shifted to the top right corner for the out the money options. Its form is also different because the correlations and variances are. The adaptive method has been able to detect this frontier automatically in quite different situations.

4.2.2 Examples in dimension 3 and 4

Our first example is a basket option on 3 uncorraleted assets with the parameter values

$$T = 3, r = 0.05, a_1 = a_2 = a_3 = 30, \sigma_1 = \sigma_2 = \sigma_3 = 0.2$$

and two strike prices $K_1 = 90$ and $K_2 = 120$. Our error indicator is based on $E_{3,15,20,R}(f)$ and we do 2000 steps of the algorithm, the first hundred ones with the FA method and the remaining ones with the NA method. This corresponds to roughly 2.7×10^6 function evaluations. We give in table 5, the values of the truncated estimations for $B_2 = 12$ and $B_2 = 13$.

We observe that our method is still better than QMC integration. The accuracy is about 6 digits on the at the money option and 5 on the one out of the money. Our second example is a basket option on 4 uncorrelated assets with

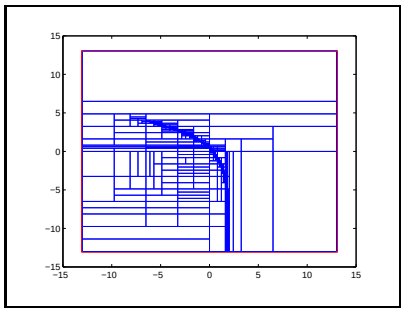

Figure 9: Ex. 1: Put for K_1

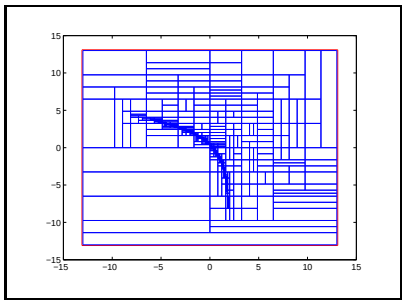

Figure 10: Ex. 1: Call for K_1

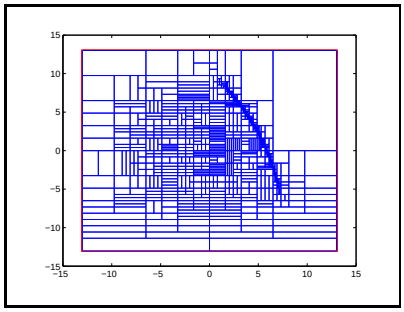

Figure 11: Ex. 2: Put for K_3

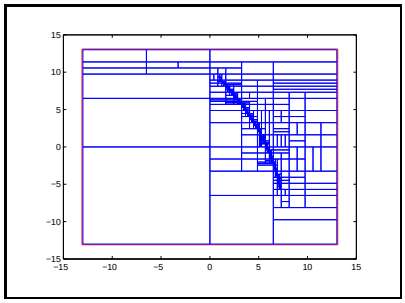

Figure 12: Ex. 2: Call for K_3

$\sigma = 0.2$	$V(T,K,B)$	$U(T,K,B)$	$C(T,K,B)$	$H(T,K,B)$
$(K,B) = (K_1, B_1)$	14.80805595	2.27176971	4×10^{-6}	5×10^{-4}
$(K,B) = (K_1, B_2)$	14.80805719	2.27177433	7×10^{-7}	6×10^{-4}
$(K,B) = (K_2, B_1)$	2.92705764	16.21209700	8×10^{-5}	9×10^{-4}
$(K,B) = (K_2, B_2)$	2.92705238	16.22195677	5×10^{-5}	1×10^{-3}

Table 5: Option pricing on 3 assets

$$T = 1, r = 0.05, a_1 = a_2 = a_3 = a_4 = 20, \sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = 0.1$$

and two strike prices $K_1 = 80$ and $K_2 = 90$. Our error indicator is based on $E_{4,15,20,R}(f)$ and we do 2000 steps of the NA method. This corresponds to roughly 8.2×10^6 function evaluations. We give in table 6, the values of the truncated estimations for $B_1 = 5$ and $B_2 = 6$. On this last example, our algorithm is comparable to QMC integration: it is slightly better on the at the money options and slightly worse on the out of the money options.

We can conclude that our adaptive algorithm is really efficient in dimensions 2 and 3 but only satisfactory in dimension 4. It might be possible to improve it by building better quadratures based on quantization points or improved Halton

$\sigma = 0.1$	$V(T,K,B)$	$U(T,K,B)$	$C(T,K,B)$	$H(T,K,B)$
$(K,B) = (K_1, B_1)$	4.2282871	0.3266767	3×10^{-5}	2×10^{-4}
$(K,B) = (K_1, B_2)$	4.2287939	0.3266880	4×10^{-4}	5×10^{-4}
$(K,B) = (K_2, B_1)$	0.1684123	5.7797621	7×10^{-4}	5×10^{-5}
$(K,B) = (K_2, B_2)$	0.1684173	5.7762449	6×10^{-4}	6×10^{-5}

Table 6: Option pricing on 4 assets

sequences. Another possibility is to make some coupling between our method and other variance reduction methods like for example the one developed in [7].

5 Conclusion

The adaptive approximation algorithm proposed in this paper is very satisfactory on most examples studied. The FA version has given very accurate results on the regular functions of the Genz package up to dimension 5. The partially adaptive method and the NA method have given an acceptable accuracy on continuous functions and even on discontinuous ones. The pricing of some basket options was also very accurate in dimensions ($s \leq 3$). Such accurate pricings can be useful to make a sensitivity analysis of the parameters of the Black-Scholes model. Nevertheless some convergence problems still remain and it is obviously hard to deal efficiently with all kind of integration problems using the same algorithm. We have focused here mainly on numerical integration and not on the piecewise multivariate polynomial approximation that we have obtained. This approximation can be very useful for the numerical resolution of partial differential equations like the Boltzmann equation or also in the computer experiment problems.

References

[1] E. I. ATANASSOV, I. T. DIMOV, A new optimal Monte Carlo method for calculating integral of smooth functions, Monte Carlo Methods and Appl., Vol. 5, No. 2, pp. 149-167, 1999.

[2] G. DEELSTRA, J. LIINEV, M. VANMAELE, Pricing of arithmetic basket options by conditioning, Insurance: Mathematics and Economics, vol 34, pp. 55-77, 2004.

[3] P. ETTORE, G. FORT, B. JOURDAIN, E. MOULINES, On adaptive stratification, To appear in Annals of operations research.

[4] A. C. GENZ, Testing multidimensional integration routines. Tools, Methods and Languages for Scientific and Engineering Computations, pp. 81-94, 1984.

[5] A. C. GENZ, A. A. MALIK, Remarks on algorithm 006: An adaptive algorithm for numerical integration over an n-dimensional rectangular region, Journal of Computational and Applied Mathematics 6 (4), pp. 81-94, 1984.

[6] P. HELLUY, S. MAIRE, P. RAVEL, Intégration numérique d'ordre élevé de fonctions régulières ou singulières sur un intervalle, CR. Acad. Sci. Paris, Sér. 1, 327, pp. 843-848, 1998.

[7] B. JOURDAIN, J. LELONG, Robust adaptive importance sampling for normal random vectors, Annals of applied probability,19 (5), pp. 1687-1718, 2009.

[8] M . H . KALOS, P. A. WHITLOCK, Monte Carlo Methods, John Wiley & Sons, 1986.

[9] A. R. KROMMER, C. W. UEBERHUBER. Computational integration. SIAM, 1998.

[10] D. LAMBERTON, B. LAPEYRE, Introduction to stochastic calculus applied to finance, Chapman & Hall, London, 1996.

[11] G. P. LEPAGE, A New Algorithm for Adaptative Multidimensional Integration, Journal of Computational Physics, Vol 27, pp. 192-203, 1978.

[12] S. MAIRE, Reducing variance using iterated control variates, The Journal of Statistical Computation and Simulation, Vol. 73(1), pp. 1-29, 2003.

[13] S. MAIRE, An iterative computation of approximations on Korobov-like spaces. Journal of Computational and Applied Mathematics, 157, pp. 261-281, 2003.

[14] S. MAIRE, Polynomial Approximations of multivariate smooth functions from quasi-random data, Statistics and Computing, 14, pp. 333-336, 2004.

[15] S. MAIRE, C. DE LUIGI, Quasi-Monte Carlo quadratures for multivariate smooth functions, Applied Numerical Mathematics, 56, no.2, pp. 146-162, 2006.

[16] H. NIEDERREITER, Quasi-Monte Carlo methods and pseudorandom numbers, Bull. Amer. Math. Soc. 84, pp. 957-1041, 1978.

[17] E. NOVAK, K. RITTER, High dimensional integration of smooth functions over cubes, Numerishe Mathematik, 75, pp.79-97, 1996.

[18] G. PAGES, A space vector quantization for numerical Integration, Journal of computational and applied mathematics, 89, pp. 1-38, 1997.

[19] W. H. PRESS, G. R. FARRAR, Recursive Stratified Sampling for Multidimensional Monte Carlo Integration, Computer in Physics, vol. 4, pp. 190-195, 1990.

[20] R. SCHURER, Adaptive quasi-Monte Carlo integration based on MISER and VEGAS. Monte Carlo and quasi-Monte Carlo methods 2002, pp. 393-406, Springer, Berlin, 2004.

[21] R. SCHURER, A comparison between (quasi)-Monte Carlo and cubature rule based methods for solving high-dimensional integration problems, Mathematics and computers in simulation, 62, pp. 509-517, 2003.

[22] I. H. SLOAN, P. J. KACHOYAN, Lattice methods for multiple integration: Theory, error analysis and examples, SIAM J. Numer. Anal. 24, pp. 116-128, 1987.